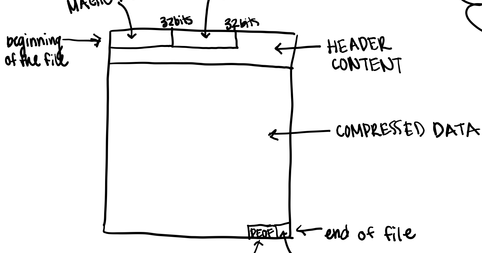
Compressed file

4/19/20 14:39

why? we want to make sure this is actually a huffman compressed file before we start decompression. since an int is 32 bits there is a $\frac{1}{2^{32}}$ chance we decompress a file that is not actually a huffman file. That's a 0.00000000232 8% chance!

why? we need to know what type of header content is coming so we can rebuild our tree correctly



32 bits = integer size

*use BITS_PER_INT constant provided

why? tells us to stop decompression

extra padding of 0's (your computer will do this for you)
why? files have to be in 8 bit (1 byte) increments. Depending on the encoding, we might be missing a few bits at the end so the computer will add on some 0's for us.

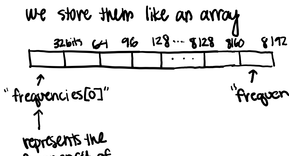
HEADER FORMATS:

COUNTS FORMAT:

there are 256 different values we can encode
($2^8 = 256$)
(bits per byte)

to remake the tree, we need to know the frequency of each value.

we store them like an array



Note: we do not include the PEOF because that frequency is always 1, you will have to add it yourself to the priority queue when decompressing

TREE FORMAT:

instead of storing frequencies (because this takes quite a bit of memory when a lot of values might not even be in the original file i.e. freq of 0) we can pass along the tree!

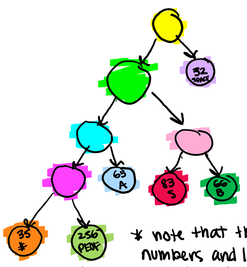
we use 0 to represent a non-leaf node
we use 1 to represent a leaf node
↳ a leaf node represents a value that was compressed after we store a 1 for the leaf node, we use 9 bits to to store the value (not freq) of that tree node
why 9? we have 256 values + 1 PEOF which means there are 257 values which fits into a minimum of 9 bits

the nodes are stored using a pre-order traversal

ex. 0 0 0 0 1 00 01000011 1 1000000000 1 001000001 0 1 00101001 1 001000010 1 000100000

Steps to read: (use recursion)
- read in 1 bit
- if 0 it is inner leaf node
  - make empty node
  - set left child to recursive call
  - set right child to recursive call
  - return self
- if 1 it is leaf node
  - read in 9 more bits (this is value of node)
  - make new node w/ this value
  - return this
- if it is -1
  throw exception

should not happen if header is formed correctly



* note that the numbers and letters in each node each represent the value of the node. They mean the same thing

not in the ascii table. we made this up!

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	(space)	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Extended ASCII Codes

128	Ç	144	É	160	á	176	░	192	└	208	╨	224	Ó	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Ô	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	ò	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Õ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	õ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	þ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Þ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Ú	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	û	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	ù	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	²	252	³
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	Ý	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ý	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	¯	255	